day01

学习目标:

- **了解常用浏览器**
- **掌握WEB标准**
- **常用HTML标签介绍**
- **掌握H5新增表单和表单属性**
- **掌握H5新增的属性**
- **CSS层叠样式表**
- **CSS中的盒子模型**
- **CSS中常用的属性介绍**

一.认识网页

网页主要由文字、图像和超链接等元素构成。当然，除了这些元素，网页中还可以包含音频、视频以及Flash等。

思考： 网页是如何形成的呢?

1.1 常见浏览器介绍

浏览器是网页运行的平台，常用的浏览器有IE、火狐（Firefox）、谷歌（Chrome）、Safari和Opera等。我们平时称为五大浏览器。

查看浏览器占有的市场份额（知晓）

查看网站： http://tongji.baidu.com/data/browser

1.2 Web标准（重点）

通过以上浏览器的不同，我们知道他们工作原理、解析肯定不同，显示就会有差别。

Web 标准构成

Web标准不是某一个标准，而是由W3C和其他标准化组织制定的一系列标准的集合。主要包括结构（Structure）、表现（Presentation）和行为（Behavior）三个方面。

- **结构标准：**结构用于对网页元素进行整理和分类，**主要指HTML**
- **样式标准**：表现用于设置网页元素的版式、颜色、大小等外观样式，**主要指的是CSS。**
- **行为标准：**行为是指网页模型的定义及交互的编写，**主要包括JavaScript**

Web 标准的好处

1、让Web的发展前景更广阔
2、内容能被更广泛的设备访问
3、更容易被搜寻引擎搜索
4、降低网站流量费用
5、使网站更易于维护
6、提高页面浏览速度

WEB标准的表现形式：

二.HTML

2.1 前端开发工具介绍

•推荐开发工具

–HBuilder（非常好用的国产 IDE）

–https://www.dcloud.io/hbuilderx.html

–WebStorm

–https://www.jetbrains.com/webstorm/

•Pycharm 集成开发工具

•官网地址：https://www.jetbrains.com/

•使用介绍

•安装及插件包

•快捷键汇总

2.2 HTML 初识

一般先学习HTML+CSS， 这里我们先定一个小目标，先学HTML,后学习CSS。

HTML（英文Hyper Text Markup Language的缩写）中文译为“超文本标签语言”，主要是通过HTML标签对网页中的文本、图片、声音等内容进行描述。

–超文本：就是指页面内可以包含图片、链接，甚至音乐、程序等非文字元素 、

–标记（ Markup ）： 成对的标签

–标记语言：标签不需编译，可以直接被浏览器解读

–后缀：.html .htm

IDEA创建HTML文件

•HTML文档结构介绍

•创建（htm和html两种后缀名的方式创建网页）+快捷键Tab

•文档声明

•Head：所有头部标签的容器，资源的引入

•Body：网页主题结构

文本编辑及运行

•我的第一个页面

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>hello</title>
</head>
<body>
    <h1>这是我创建的一个网页</h1>
</body>
</html>
```

HTML标签分类

在HTML页面中，带有“< >”符号的元素被称为HTML标签，如上面提到的 <HTML>、<head>、<body> 都是HTML标签。所谓标签就是放在“< >” 标签符中表示某个功能的编码命令，也称为HTML标签或HTML元素

1.双标签

```
<标签名> 内容 </标签名>
```

该语法中“<标签名>”表示该标签的作用开始，一般称为“开始标签（start tag）”，“</标签名>” 表示该标签的作用结束，一般称为“结束标签（end tag）”。和开始标签相比，结束标签只是在前面加了一个关闭符“/”。

> 比如 <body>我是文字 </body>

2.单标签

```
<标签名 />
```

单标签也称空标签，是指用一个标签符号即可完整地描述某个功能的标签。

> 比如

HTML标签关系

标签的相互关系就分为两种：

1.嵌套关系

```
<head>  <title> </title>  </head>
```

2.并列关系

```
<head></head>
<body></body>
```

测试题：

```
请问下列哪个标签是错误的？
```

```
A  <head></head><body></body>
```

```
B  <strong><div></div></strong>
```

```
C  <head><title></head></title>
```

```
D  <body><div></div></body>
```

2.3 HTML中常用的标签

h标签，标题标签

- 用法：自动换行功能，加粗（注意最多只有六级）
- 作用：重要因为爬虫在爬取内容时会根据h标签来理解文档的结构
- 场合：标题标签
- 案例练习

```

```

水平线标签(认识)

单词缩写： horizontal 横线

在网页中常常看到一些水平线将段落与段落之间隔开，使得文档结构清晰，层次分明。这些水平线可以通过插入图片实现，也可以简单地通过标签来完成，

就是创建横跨网页水平线的标签。其基本语法格式如下：

```
<hr />是单标签
```

在网页中显示默认样式的水平线。

P标签，段落标签

- 单词缩写： paragraph 段落
- 用法：主要是文档内容段落，注意转义字符：空格（ ），大于（&gt）小于（&lt）
- 作用：段落
- 场合: 段落标签
- 案例练习

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>标题</title>
</head>
<body>
<h1>这是一级标题</h1>
<h2>这是二级标题</h2>
<h3>这是三级标题</h3>
<h4>这是四级标题</h4>
<h5>这是五级标题</h5>
<h6>这是六级标题</h6>

<!--可以用户字符实体解决空格，大于号和小于号等转义字符的一些问题-->
<!--空格： 
    小于：&lt;
    大于：&gt;-->
<p>    nsajdjsaidsaojdffisamcsakmcsamcascsoaaojx
jsdkmsadkmsalkdmadmakldmsalkmlka</p>
<p> 3 &lt; 5</p>
<p> 8 &gt; 5</p>

<!--换行标签br-->
东方瑞通<br>
您终身的学习伴侣
</body>
</html>
```

常用的转义字符

显示结果	描述	实体名称	实体编号
	空格		
<	小于号	&lt;	&#60;
>	大于号	&gt;	&#62;
&	和号	&amp;	&#38;
"	引号	&quot;	&#34;
'	撇号	' (IE不支持)	&#39;
¢	分	&cent;	&#162;
£	镑	&pound;	&#163;
¥	人民币/日元	&yen;	&#165;
€	欧元	&euro;	&#8364;
§	小节	&sect;	&#167;
©	版权	&copy;	&#169;
®	注册商标	&reg;	&#174;

br标签（单标签：不需要结尾），可以有结束，也可以没有结束

- 用法：需要换行标签的末尾添加
- 作用：换行标签
- 场合：换行

块级标签div，span

div span　是没有语义的　是我们网页布局主要的2个盒子

div 就是 division 的缩写　分割，分区的意思 其实有很多div 来组合网页。

span, 跨度，跨距；范围

- 用法：div表示一块内容没有具体含义，定义后可在这一块区域内添加任何标签，Span标签本身没有任何语义，不过我们可以借用span标签给某段文字加样式
- 作用：不带有任何样式的块级标签，定义某一块区域
- 场合: 块级标签，可以定义一块内容，区域内可以放入多个标签

em标签，i标签，b标签，strong标签

标签	显示效果
	文字以粗体方式显示（XHTML 推荐使用 strong）
和	文字以斜体方式显示（XHTML 推荐使用 em）
<s></s>和<del></del>	文字以加删除线方式显示（XHTML 推荐使用 del）
<u></u>和<ins></ins>	文字以加下划线方式显示（XHTML 不赞成使用 u）

img标签

- 用法：在文档中添加图像显示
- 作用：图片显示，src属性和alt(定义图片加载失败时显示的文字，搜索引擎会使用文字收录图片，盲人读屏软件会读取内容帮助盲人识别图片)属性

<img />标记属性

属性	属性值	描述
src	URL	图像的路径
alt	文本	图像不能显示时的替换文本
title	文本	鼠标悬停时显示的内容
width	像素（XHTML不支持%页面百分比）	设置图像的宽度
height	像素（XHTML不支持%页面百分比）	设置图像的高度
border	数字	设置图像边框的宽度

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>图像标签</title>
</head>
<body>
<!--相对路径：相对于这个html文件的位置./是可以省略的-->
  <img src="./img/1.png" alt="飞机图片">
<!--绝对路径：-->
<img src="C:\Users\amdin\Desktop\img\2.png" alt="飞机图片">

</body>
</html>
```

a标签，超链接标签

- 用法：连接到某一个网页或者地址
- 作用：超链接
- 场合：页面跳转或者地址链接
- 案例：本地超链接，百度超链接
- Title属性：Title：鼠标悬停后提示内容
- target属性介绍：

  Target：*self(把自己替换掉，依然跳到当前页)*，blank（表示另外再打开一个页面窗口，依然去往目标页面）

注意：

1.外部链接 需要添加 http:// www.baidu.com

2.内部链接 直接链接内部页面名称即可 比如 < a href="index.html"> 首页

3.如果当时没有确定链接目标时，通常将链接标签的href属性值定义为“#”(即href="#")，表示该链接暂时为一个空链接。

4.不仅可以创建文本超链接，在网页中各种网页元素，如图像、表格、音频、视频等都可以添加超链接。

有序列表标签（ol，li）

- 用法：以列表形式显示
- 作用：有序列表
- 场合：有序列表制作（应用不是很多）
- 案例练习

无序列表标签（ul，li）

- 用法：以列表形式显示
- 作用：无序列表
- 场合: 无序列表制作（应用比较广泛）
- 案例练习

定义列表dl，dt，dd

•介绍

dl：（Definition List）定义列表

dt：（Definition Term）定义术语

dd：（Definition Description）定义描述

•用法：用于术语的定义

•作用：dl表示列表的整体，dt表示术语的标题，dd表示术语的内容描述

•案例：本地超链接，百度超链接

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>列表标签</title>
</head>
<body>

  <h3>有序列表</h3>
<!-- 有序列表-->
  <ol >
    <li>Android</li>
    <li>Java</li>
    <li>Python</li>
  </ol>

<h3>无序列表</h3>
<!-- 无序列表-->
  <ul title="列表">
    <li>Android</li>
    <li>Java</li>
    <li>Python</li>
  </ul>

<h3>定义列表</h3>
  <dl>
    <dt>html</dt>
    <dd>负责页面显示的结构</dd>

    <dt>css</dt>
    <dd>负责页面显示的效果</dd>
```

```
    <dt>javascript</dt>
    <dd>负责页面显示的行为</dd>
  </dl>


  <ul>
    <li>Python</li>
    <li>Python</li>
    <li>Python</li>
    <li>Python</li>
    <li>Python</li>
    <li>Python</li>
    <li>Python</li>
    <li>Python</li>
  </ul>

</body>
</html>
```

表格标签table

•介绍

table：（table）表格

tr：（table row）表格中的行

td：（table data cell）表格行中的列

•用法：用于页面表格数据展示

•属性：

Border边框，align对齐，colspan合并列，rowspan合并行

作用：页面表格展示

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>表格标签</title>
</head>
<body>

  <h3>表格标签</h3>

<table border="1px" width="400">
  <tr>
<!--     表示表头，字体默认加粗居中-->
    <td>name</td>
    <td>age</td>
    <td>gender</td>
    <td>class</td>
  </tr>
  <tr align="center">
    <td>张三</td>
```

```
    <td>18</td>
    <td>男</td>
    <td>python</td>
  </tr>
  <tr align="center">
    <td>李四</td>
    <td>19</td>
    <td>女</td>
    <td>UI</td>
  </tr>

  <tr>
    <!--占用4列-->
    <td colspan="4">总计</td>
  </tr>
</table>

<h3>表格标签</h3>

<table border="1" width="600" height="300" align="center">
  <tr align="center">
    <td colspan="5">个人简历</td>
  </tr>
  <tr align="center">
    <td width="20%">姓名</td>
    <td width="20%"></td>
    <td width="20%">性别</td>
    <td width="20%"></td>
    <td rowspan="4" width="20%"><img src="icon.png"></td>
  </tr>
  <tr align="center">
    <td>民族</td>
    <td></td>
    <td>出生日期</td>
    <td></td>

  </tr>
  <tr align="center">
    <td>政治面貌</td>
    <td></td>
    <td>健康状况</td>
    <td></td>

  </tr>
  <tr align="center">
    <td>籍贯</td>
    <td></td>
    <td>学历</td>
    <td></td>

  </tr>
</table>

<table>
  <tr>
    <td></td>
    <td></td>
    <td></td>
```

```
    <td></td>
  </tr>
  <tr></tr>
  <tr></tr>
  <tr></tr>
</table>

</body>
</html>
```

Form表单标签

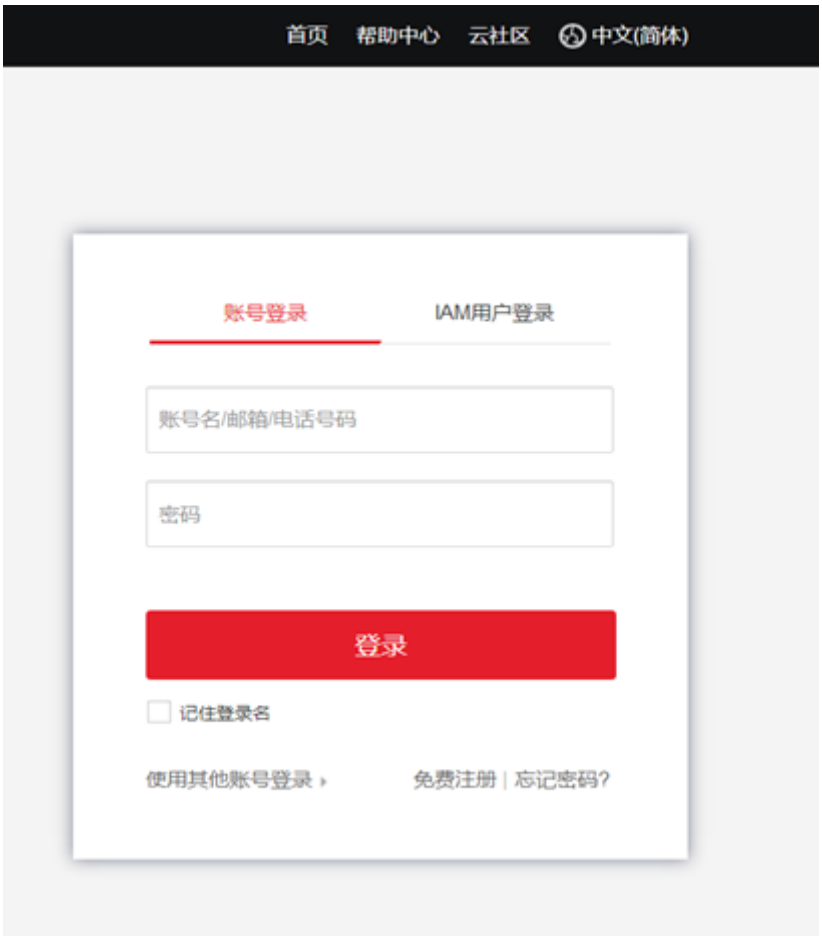
•用法：表单

•作用：定义整体的表单区域

•属性介绍：

Action：定义表单提交的地址

Method：定义表单提交的方式：get和post两种方式




```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>
<!--form表单，主要用来实现数据以表单形式提交给后端，实现前后端的交互
    action：就是数据提交的地址
```

```
    method：提交数据方式：
        get请求：数据在地址栏拼接的方式提交，默认就是get
        post请求：数据以数据流的形式写入到后端

    type：设置表单的格式
        submit：form表单在提交数据时必须是submit才可以把数据提交到后端
        reset：重置，把表单中内容置空-->
    <form action="#" method="post">
        UserName:<input type="text" name="username"><br>
        PassWord:<input type="password" name="password"><br>
        <input type="submit" value="登录">
        <input type="reset" value="重置">
    </form>

</body>
</html>
```

Type属性值

•Text：定义单行可输入文本框

•Password：定义密码输入框

•Radio：定义单选框

•Checkbox：定义复选框

•Email：定义邮箱格式输入框

•File：定义文件上传

•Submit：定义提交按钮

•Reset：定义重置按钮

H5新增表单控件属性

•placeholder：占位符，隐藏提示

•autofocus：自动获取焦点，不用用户定焦点

•Multiple：上传多个文件

•Autocomplete：设置表单是否自动完成，具有记忆功能，自动记录完成

•必须要有提交按钮

•必须要有name属性

•Required：此项不能为空，必填项

•accesskey=“s”：设置获取焦点的快捷键为alt+s

Textarea标签

•用法：在表单中定义一块文本区域

•作用：表单中个人介绍等片区文本输入

•场合：表单提交中

•属性：rows：文本区行数，cols：文本区列数

Select，option标签

•用法：下拉选择框

•作用：主要用于下拉选择框

•场合：省市联动等

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>
<form action="">
    <!--required:设置为必填项-->
    <!--autofocus：设置表单自动获取焦点-->
    <!--accesskey：设置表单通过快捷键的方式获取焦点-->
    <input type="text" placeholder="请输入用户名" name="username" required
accesskey="s"><br>
    <input type="password" placeholder="请输入密码" name="password" required><br>
    性别：<input type="radio" name="gender">男<input type="radio" name="gender">女
<br>
    爱好：<input type="checkbox">抽烟<input type="checkbox">喝酒<input
type="checkbox">烫头<br>
    邮箱：<input type="email"><br>
    照片：<input type="file" multiple><br>
    个人介绍：
    <textarea name=""  cols="30" rows="10"></textarea><br>
    个人专业：
    <select name="">
        <option value="">---请选择---</option>
        <option value="">计算机专业</option>
        <option value="">电子专业</option>
        <option value="">金融</option>
    </select>
    <br>
    <input type="submit" value="注册">
    <input type="reset" value="重置">
</form>
</body>
</html>
```

Datalist标签

类似百度搜索框输入后的下拉，定义选项列表，一般和input标签联合使用：

•案例：明星

Fieldset

将表单里面的元素分组，打包。一般和legend（标题）搭配使用

•案例：用户登录

```
<!DOCTYPE html>
<html lang="en">
```

```
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>

<!--分组：表示是同一块区域中的额内容可以作为一个分组-->
<fieldset style="width: 400px">
    <!--给分组添加标题-->
    <legend>用户登录</legend>
    <form action="">
        <input type="text" name="username" placeholder="用户名"><br>
        <input type="password" name="password" placeholder="密码"><br>
        <input type="datetime-local">
        <input type="submit" value="登录">
        <input type="button" value="注册">
    </form>

</fieldset>

    <!--定义选项列表，input输入框和列表之间通过id和list属性建立连接-->
    <input type="text" list="Star">
    <datalist id="Star">
        <option value="刘德华"></option>
        <option value="刘若英"></option>
        <option value="刘亦菲"></option>
        <option value="岳云鹏"></option>
        <option value="name"></option>
        <option value="nameset"></option>
    </datalist>

</body>
</html>
```

三. CSS样式

CSS为层叠样式表

作用

•Html只负责文档的结构和内容，表现形式完全交由css负责

•页面样式渲染

•让html文档变得简洁，方便爬虫去爬去界面内容，因为爬虫只爬去html文档中的内容，不会爬去css样式

3.1 添加CSS样式的方式

- 方式一：

  直接在标签内部添加style属性后添加样式

- 方式二：

  直接在html文档的head头部标签内添加style标签

- 方式三：

  直接在html文档外部新建.css后缀的样式文件，后使用link标签在head标签内引入

代码展示：

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <!--引入外部css样式,link标签，通过href属性引入-->
    <link rel="stylesheet" href="../css/test01.css">
    <style>
        div{
            width: 200px;
            height: 100px;
            background-color: deepskyblue;
            /*设置div中的文本水平垂直居中*/
            text-align: center;
            /*垂直居中，通过设置行高的方式*/
            line-height: 100px;
        }
    </style>
</head>
<body>
    <!--給标签添加css样式有三种方式：-->
    <!--方式一：直接在标签内部，通过style属性添加样式-->
    <h2 style="color: deeppink;text-decoration: underline">这是一个标题</h2>

    <!--方式二：在head中添加style，在style标签中添加样式-->
    <div>这是一个div</div>

    <!--方式三：在外部创建css文件，写入样式，在引入到当前文件中，通过link在head中引入-->
    <a href="http://www.baidu.com" target="_blank">百度一下</a>
</body>
</html>
```

CSS常用样式

•Color

•设置文字颜色

•Font-size

•设置文字大小

•Font-family

•设置文字的字体

•Font-style

•设置字体是否倾斜，normal不倾斜，italic倾斜

•Font-weight

•设置字体是否加粗，normal不加粗，bold加粗

•Line-height

•设置行高，相当于在每行文字的上下各加间距使文字垂直居中

•Font

•可以综合同时设置几个属性：是否加粗，字号，行高，字体的顺序

•text-decoration

•设置文字下划线

•text-indent

•设置文字前缩进，一般用于首行缩进，缩进距离是字体大小的2倍

•Text-align

设置文本对齐方式

3.2 CSS样式选择器

•*号选择器

•选择HTML文档中的所有标签

•标签选择器

•选择跟标签名相同的标签

•Id选择器

•选择标签中id属性名

•特性：页面上标签的id名是唯一不能够重复的（局限性）

•Class类选择器

•Css中最常用的一种选择器，class属性是可以重复使用在不同的标签上

一个选择器可以应用到多个标签上，一个标签上也可以使用多个选择器

•层级选择器

•主要应用在选择父元素下的子元素或者子元素下的子元素

•非常常用，一般标签选择器不单独使用，会和层级选择器一起用

•注意：层级选择器一般不要超过四层，不然会影响页面响应效果，影响性能

•组选择器

•一般多个选择器有同样的样式可以使用组选择器

•组选择器的每个选择器中间用，号隔开

•伪类及伪元素选择器

•伪类hover：一般用于连接的响应式效果，可以修改鼠标悬停样式

•伪元素选择器：before在标签内容前面添加内容，after在标签内容后面添加内容，注意添加的内容都是不能被 选中的

```
<!DOCTYPE html>
<html lang="en">
<head>
```

```
    <meta charset="UTF-8">
    <title>Title</title>

    <style>
        /*通配符选择器：（*），可以选中html文档中的所有标签,优先级最低，为0*/
        *{
            color: red;
        }
        /*标签选择器：直接使用标签名的方式添加样式，可以选中整个html文件中的所有标签名的标签
*/
        /*需要同时选中整个文档中所有标签名一样的标签*/
        h2{
            text-decoration: underline;
            font-style: italic;
            font-weight: normal;
        }
    /*只给第二个div添加某个样式，在整个html文件中所有标签中的id属性值是唯一的不能重复的*/
        /*使用id选择器，使用#+id名的方式*/
        /*缺点：只能够选中一个标签来添加样式，一般情况我们不使用id选择器给标签添加样式*/
        #div02{
            width: 100px;
            height: 100px;
            border: 1px solid blue;
        }
        /*class类选择器：具有相同样式的标签的类属性值可以设置成一样的，方便设置样式*/
        /*.加类名的方式*/
        .list1 .single{
            background-color: gold;
        }
        .list1 .double{
            background-color: hotpink;
        }
        .list1{
            /*去掉无序列表原生的样式*/
            list-style: none;
        }

        /*给div01和div03添加相同的样式，可以使用组选择器，具有相同样式的划分为一组*/
        #div01,#div03{
            width: 200px;
            height: 100px;
            background-color: deepskyblue;
        }

        /*层级选择器：针对父级标签嵌套子级标签*/
        .list2 .single{
            background-color: deepskyblue;
        }

        /*伪类选择器hover：设置鼠标悬停的效果*/
        #div02:hover{
            background-color: aqua;
            color: gold;
            text-align: center;
            line-height: 100px;
        }
    </style>
</head>
```

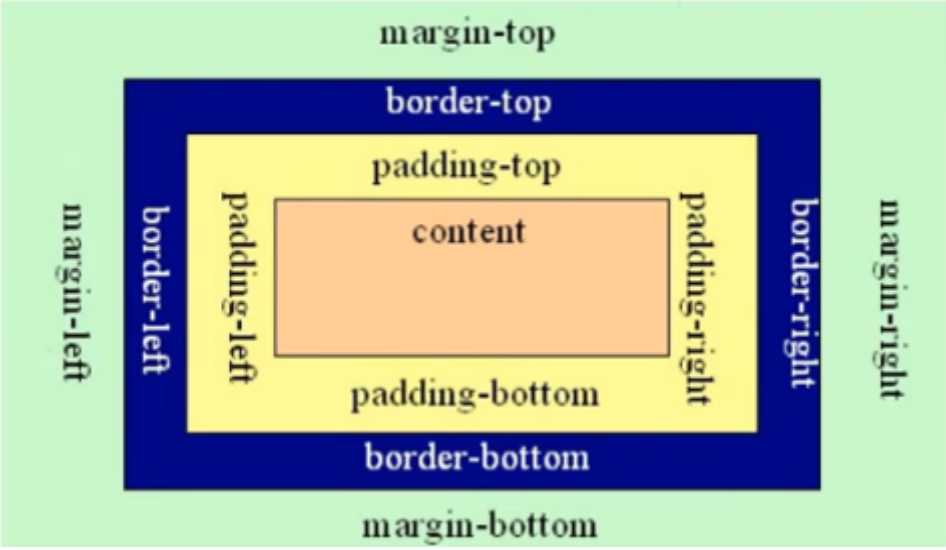
```
<body>
    <!--css中额选择器：通过选择器我们可以指定标签的样式，七种选择器-->
    <h2>这是一个标题</h2>
    <h2>这是另外一个标题</h2>

    <div id="div01">这是div01</div>
    <div id="div02">这是div02</div>
    <div id="div03">这是div03</div>

    <a href="#">点我有惊喜！</a>

    <!--给奇数添加相同的样式，给偶数添加相同额样式，使用类选择器-->
    <ul class="list1">
        <li class="single">速度与激情1</li>
        <li class="double">速度与激情2</li>
        <li class="single">速度与激情3</li>
        <li class="double">速度与激情4</li>
        <li class="single">速度与激情5</li>
        <li class="double">速度与激情6</li>
    </ul>

    <ul class="list2">
        <li class="single">1</li>
        <li class="double">2</li>
        <li class="single">3</li>
        <li class="double">4</li>
        <li class="single">5</li>
        <li class="double">6</li>
    </ul>
</body>
</html>
```

案例：新闻标题

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>

    <style>
        .list{
            list-style: none;
        }
        .list li{
            width: 100px;
            height: 50px;
            border: 1px solid red;
            text-align: center;
            line-height: 50px;
            background-color: deepskyblue;
        }
        /*伪类选择器*/
        .list li:hover{
            background-color: hotpink;
            color: blue;
        }
```

```
    </style>
</head>
<body>

    <ul class="list">
        <li>新闻标题1</li>
        <li>新闻标题2</li>
        <li>新闻标题3</li>
        <li>新闻标题4</li>
        <li>新闻标题5</li>
        <li>新闻标题6</li>
    </ul>

</body>
</html>
```

3.3 盒子模型



•盒子的基本设置

•Width宽，height高

•Border属性设置

•Border-width：边框宽，border-style：边框样式，border-color：颜色

•Border综合写法，三个属性一同写入

•Padding属性设置

•Padding-top上内边距，right，bottom，left

•综合写法：顺时针设置属性，三个值（上，左右，下）两个值（上下，左右）

•Margin属性设置

•Margin顺时针写法

•注意：margin属性独有的，可以使元素水平居中

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>

    <style>
        #box{
            height: 100px;
            width: 100px;
            /*给盒子添加边框*/
            /*边框是计算在盒子大小里面的*/
            border: 5px solid red;
            /*给盒子添加内边距padding*/
            /*内边距也是计算在盒子大小里面的*/
            padding: 10px;
            /*从top开始，按照顺时针方向旋转*/
            /*padding: 10px 5px 8px 7px;*/
            /*给盒子添加外边距margin*/
            /*外边距是不计算在盒子大小里面的*/
            margin: 10px;
            background-color: deepskyblue;
        }
    </style>
</head>
<body>

<!--盒子模型：盒子的大小计算-->

<div id="box">这是div</div>

</body>
</html>
```

结论：盒子的大小包含边框，内边距的大小，不包含外边距margin的大小

3.4 CSS中常用属性

margin的扩展属性：

•Auto

•设置后可以实现元素水平方向居中，注意auto属性只能用于水平方向

•案例展示

•Margin设置负值

•盒子实现位移（反方向），注意body标签在浏览器上本身有8px边距

•案例展示

•Margin边距合并

•同一方向的两个元素都添加margin的话只会执行那个数值大的（只会执行一个）

•案例展示

•Margin顶部塌陷

•出现在另个盒子嵌套使用时，内部盒子添加margin-top属性时会自动传递给外部盒子，而内部盒子的margin- top 不起作用

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>

    <style>
        #box{
            width: 100px;
            height: 100px;
            background-color: deepskyblue;
            /*通过margin属性设置盒子水平居中*/
            margin: 50px auto;
        }

        #box02{
             width: 100px;
            height: 100px;
            background-color: deeppink;
            /*margin设置为负值表示向反方向移动*/
            margin-left: -30px;
            margin-bottom: 50px;
        }
        #box03{
            width: 100px;
            height: 100px;
            background-color: burlywood;
            /*margin的边距合并，给两个盒子在同一个方向设置两个margin的那时候
            他只会执行那个值比较大的margin*/
            margin-top: 20px;
        }
        .big{
            width: 300px;
            height: 300px;
            background-color: gold;
            border: 1px dashed red;
        }
        .small{
            width: 100px;
            height: 100px;
            background-color: deepskyblue;
            /*margin的顶部塌陷，盒子嵌套时，内部盒子添加margin-top属性时会自动传递给外部盒子，而内部盒子的margin-top不起作用*/
            /*给大盒子添加边框解决*/
            margin-top: 30px;
        }
    </style>
</head>
<body>


    <div id="box">div01</div>
```

```
    <div id="box02">div02</div>
    <div id="box03">div03</div>

    <div class="big">
        <div class="small"></div>
    </div>

</body>
</html>
```

盒子的分类（元素的分类）

- **块级元素**

  •概念：块元素也可以称为行元素，整个元素定义后占用一行

  •常见块元素标签：div，p，ul，li，h1-h6，dl，dt，dd等

  •特性：

  支持全部的样式

  如果没有设置宽度，默认宽度为父级宽度的100%

  独自占据一行，即使设置了宽度

- **内联元素**

  •概念：内联元素也可以称为行内元素，元素不会独自占据一行

  •常见内联元素标签：img,a，span，em，b，strong，i等

  •特性：

  支持部分样式（不支持宽高，margin上下）

  宽高由内容决定

  盒子并在一起

  元素不换行，盒子之间会产生间隙

  子元素是内联元素，父元素可以使用text-align属性设置子元素水平对齐方式

- **内联块元素**

  •概念：内联块元素也可以称为行内块元素，元素独自占据一行

  •内联块元素师新增的元素类型，没有特别划分，我们可以通过设置来把块元素和内联元素转化为内联块元素

  •好处：不仅解决了块元素独占一行的问题，也解决了内联元素不能设置宽高的问题

  •特性：

  支持全部样式

  不设置宽高时，宽高由内容决定

  盒子并在一起

  元素换行，盒子之间会产生间隙

  子元素是内联元素，父元素可以使用text-align属性设置子元素水平对齐方式

- **元素之间的转换**

  •display

  用法：用来设置元素的类型及隐藏及隐藏

•常用值

None：元素隐藏，且不占用位置

Block：元素设置为块元素类型

Inline：元素设置为内联元素类型

Inline-block：元素设置为内联块元素类型

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>

    <style>
        .box01{
            background-color: deepskyblue;
        }
        .a1{
            background-color: deeppink;
            color: blue;
            /*高度和宽度不起作用*/
            /*把内联元素转换成块级元素借助display*/
            /*block：块级标签，inline：内联标签，inline_block:内联块*/
            display: block;
            width: 100px;
            height: 50px;
            margin-top: 40px;
            margin-left: 20px;
        }
        .news{
            font-size: 0px;
        }
        .news a{
            /*转换成块级标签*/
            display: inline-block;
            width: 100px;
            height: 50px;
            border:1px solid blue;
            text-align: center;
            line-height: 50px;
            background-color: deepskyblue;
            font-size: 20px;
        }
        .news a:hover{
            background-color: hotpink;
        }

    </style>
</head>
<body>

    <!--盒子的分类：1.块级元素，2.内联元素（行内），3.内联块元素-->

    <div class="box01">这是div01</div>

    <a class="a1" href="#">百度一下</a>
```

```
    <div class="news">
        <a href="">新闻标题1</a>
        <a href="">新闻标题2</a>
        <a href="">新闻标题3</a>
        <a href="">新闻标题4</a>
        <a href="">新闻标题5</a>
        <a href="">新闻标题6</a>
    </div>
</body>
</html>
```

float浮动

•只能设置元素左右浮动

•特性：

•Float：left，float：right

•相邻浮动的块元素可以并在一行，超出父级宽度就换行

•浮动可以让行内元素和块元素自动转化为行内块元素（并且不会有行内块元素间隙问题）

•浮动元素后面没有浮动的元素，那么没有浮动的元素中的文字会避开浮动元素的位置，形成文字绕图

•父元素如果没有设置尺寸，那么父元素内整体浮动的子元素无法撑开父元素，这时父元素需要清除浮动

•浮动元素之间没有垂直margin的合并

清楚浮动的三种方式

•方式一：父元素添加属性：overflow；hidden

•方式二：在最后一个子元素后面添加一个div，给他样式属性为clear：both（清除两端）不推荐使用

•方式三：使用伪元素after，clearfix类添加到父级元素即可，具体代码：

•Content：""，display:table，clear:both

•最终既要解决margin-top顶部塌陷，又要解决清除浮动，代码经常这样写

•Zoom：1解决的是ie浏览器不识别after属性的问题，ie现在叫做非标准浏览器

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>

    <style>
        .list{
            list-style: none;
        }
        .list li{
            height: 50px;
            width: 100px;
            border: 1px solid red;
            background-color: hotpink;
```

```
            text-align: center;
            line-height: 50px;
            /*水平浮动
            left:以左边为基准向右浮动
            right：以右边为基准向左浮动*/
            float: right;
        }

        .big{
            /*小盒子不会把大盒子撑起来，原因是小盒子设置浮动后漂浮在大盒子的上方*/
            /*清除浮动*/
            width: 400px;
            /*height: 400px;*/
            border: 1px solid red;
            margin: 50px auto;
            /*清除浮动，直接在大盒子中设置属性即可*/
            overflow: hidden;
        }
        .big div{
            height: 100px;
            width: 100px;
            background-color: deepskyblue;
            float: left;
            margin: 15px;
        }

    </style>
</head>
<body>

<!--<ul class="list">-->
<!--    <li>新闻标题1</li>-->
<!--    <li>新闻标题2</li>-->
<!--    <li>新闻标题3</li>-->
<!--    <li>新闻标题4</li>-->
<!--    <li>新闻标题5</li>-->
<!--    <li>新闻标题6</li>-->
<!--</ul>-->


    <div class="big">
        <div>1</div>
        <div>2</div>
        <div>3</div>
        <div>4</div>
        <div>5</div>
        <div>6</div>
        <div>7</div>
        <div>8</div>
        <div>9</div>
    </div>

</body>
</html>
```

定位：position

•文档流概念（为什么使用定位？）：文档流就是盒子在文档中排列的顺序，默认是从左向右，从上向下排列的，如果要改变盒子的位置，这个时候就涉及到定位了

•定位方式

•相对定位position：relative

•绝对定位position：absolute

•固定定位position：fixed

- **相对定位**

  •原理：生成相对定位的元素，元素所占据的文档流的位置保留，元素本身相对于自身的原来位置进行偏移

  •定位一般和float属性联合使用

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        .big{
            height: 300px;
            width: 300px;
            border: 1px solid red;
            margin: 0 auto;
            /*在父容器中设置定位的属性，可以让子容器以这个父容器为基准进行定位*/
            position: relative;
        }
        .big .small01{
            width: 200px;
            height: 100px;
            background-color: deepskyblue;
            /*相对定位修改small01的位置*/
            /*定位后元素文档流位置保留，是以自己原来的位置为基准进行定位的*/
            position: relative;
            left:50px;
            top: 50px;

        }
         .big .small02{
            width: 200px;
            height: 100px;
            background-color: deeppink;
        }
    </style>
</head>
<body>

<!--定位-->
<div class="big">
    <div class="small01"></div>
    <div class="small02"></div>
</div>
```

```
</body>
</html>
```

- **绝对定位**

  •原理：生成绝对定位的元素，元素脱离文档流，不占用文档流的位置，相当于漂浮在文档流的上方，定位是相对于已经设置了定位属性的父级元素定位，如果父级没有设定，那么就相对与body定位

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        .big{
            height: 300px;
            width: 300px;
            border: 1px solid red;
            margin: 0 auto;
            /*在父容器中设置定位的属性，可以让子容器以这个父容器为基准进行定位*/
            position: relative;
        }
        .big .small01{
            width: 200px;
            height: 100px;
            background-color: deepskyblue;

            /*绝对定位：absolute*/
            /*定位后元素文档流位置不占用，绝对定位默认情况下是以body标签为基准进行定位，*/
            /*如果想要以父容器为基准进行定位，需要在父容器中添加定位的属性才可以*/
            position: absolute;
            top: 20px;
            left: 20px;
        }
         .big .small02{
            width: 200px;
            height: 100px;
            background-color: deeppink;
        }
    </style>
</head>
<body>

<!--定位-->
<div class="big">
    <div class="small01"></div>
    <div class="small02"></div>
</div>

</body>
</html>
```

- **固定定位**

  •原理：生成固定定位的元素，元素脱离文档流，不占用文档流的位置，相当于漂浮在文档流的上方，定位是相对于浏览器进行定位

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        .big{
            height: 300px;
            width: 300px;
            border: 1px solid red;
            margin: 0 auto;
            /*在父容器中设置定位的属性，可以让子容器以这个父容器为基准进行定位*/
            position: relative;
        }
        .big .small01{
            width: 200px;
            height: 100px;
            background-color: deepskyblue;

            /*固定定位，以浏览器为基准进行定位*/
            position: fixed;
            top: 20px;
            right: 20px;
        }
         .big .small02{
            width: 200px;
            height: 100px;
            background-color: deeppink;
        }
    </style>
</head>
<body>

<!--定位-->
<div class="big">
    <div class="small01"></div>
    <div class="small02"></div>
</div>

<!--div{21321}*500-->

</body>
</html>
```

Static默认定位

•原理：static默认值，没有定位，元素正常出现在文档流位置，相当于取消定位属性

•注意：固定定位和绝对定位的块元素和行内元素都会转换为行内块元素

案例：微信头像

Background属性

•作用：使用比较频繁，主要用来给元素设置背景颜色和背景图片，复合属性

•属性：

- •Background-color：设置背景颜色
- •Background-image：设置背景图地址url
- •Background-repeat：设置背景图片重复方式
- •Background-position：设置背景图片的位置
- •Background-attachment:设置背景图片是否随着页面滚动而滚动
- •Background：复合设置

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>background</title>
  <style type="text/css">
    .box{
      width: 400px;
      height: 300px;
      border: 3px solid black;
      /*默认平铺
        repeat：缺省值，平铺所有
        repeat-x：平铺x
        repeat-y：平铺y
        no-repeat：只平铺一次
      */
      background-image: url("img/1.png");
      background-repeat: repeat-x;
      /*前面的是：水平方向
        后面的是：垂直方向*/
      /*background-position: left center;*/
      background-position: -30px 60px;
      /*  综合书写*/
      /*background: url("img/1.png") left 10px no-repeat red;*/

    }
  </style>
</head>
<body>

  <div class="box">
    图片
  </div>

</body>
</html>
```

CSS圆角radius和透明度rgba

•radius属性：

•Border-top-left-radius：设置某一个角的圆角

•Border-radius：顺时针同时分别设置四个角的圆角

•Border-radius：设计四个圆角相同

•Rgba透明度

•opacity: 0.6

•复合写法：background-color: rgba(0,0,0,0.4);

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>透明度</title>
  <style type="text/css">
    body{
      background: url("img/tran.png");

    }
    .box{
      height: 80px;
      width: 120px;
      color: white;
      /*background-color: #999999;*/
      text-align: center;
      line-height: 80px;
      /*之前写法：里面的字也会变得透明*/
      /*opacity: 0.6;*/
      /*!*兼容ie浏览器*!*/
      /*filter: alpha(opacity=30);*/

    /*  简易写法*/
      background-color: rgba(0,0,0,0.4);
    }
  </style>
</head>
<body>

  <div class="box">
    盒子
  </div>

</body>
</html>
```

3.4 CSS中的动画

transition动画（属性动画）

属性	描述
transition - property	指定要过渡的 CSS 属性
transition - duration	指定完成过渡要花费的时间
transition - timing-function	指定过渡函数
transition - delay	指定过渡开始出现的延迟时间

通过上述的一些属性来设置动画效果，**属性动画需要触发才会执行。**

属性详解：

- **transition-property**

不是所有属性都能过渡，只有属性具有一个中间点值才具备过渡效果。完整列表，见这个列表 ，具体效果，见 这个页面。

- **transition-duration**

指定从一个属性到另一个属性过渡所要花费的时间。默认值为0，为0时，表示变化是瞬时的，看不到过渡效果。

- **transiton-timing-function**

过渡函数，有如下几种：

liner ： 匀速
ease-in：减速
ease-out：加速
ease-in-out：先加速再减速

transition的优点在于简单易用，但是它有几个很大的局限。

（1）transition需要事件触发，所以没法在网页加载时自动发生。
（2）transition是一次性的，不能重复发生，除非一再触发。
（3）transition只能定义开始状态和结束状态，不能定义中间状态，也就是说只有两个状态。
（4）一条transition规则，只能定义一个属性的变化，不能涉及多个属性。
CSS Animation就是为了解决这些问题而提出的。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>动画</title>
  <style type="text/css">
    .box{
      height: 100px;
      width: 100px;
      background-color: hotpink;
      margin: 50px auto;
      /*如果每个动画执行时间不是同时的，可以以下方式*/
      /*transition: width 500ms ease 20ms,height 500ms ease ,border-radius 500ms
ease,background-color 500ms ease;*/

    /*  多个属性同时做动画，可以用以下方式;*/
      transition: all 500ms linear 2s;
```

```
    }
    .box:hover{
      width: 500px;
      height: 500px;
      border-radius: 250px;
      background-color: purple;
    }
  </style>
</head>
<body>

  <div class="box">

  </div>

</body>
</html>
```

复合写法：transiton: 过渡属性 过渡所需要时间 过渡动画函数 过渡延迟时间;

transform变换

变换是指从一个状态打破另一个状态的变化，常用的变化操作有平移，缩放，旋转，斜切等。

函数	描述
matrix(*n*,*n*,*n*,*n*,*n*,*n*)	定义 2D 转换，使用六个值的矩阵。
translate(*x*,*y*)	定义 2D 转换，沿着 X 和 Y 轴移动元素。
translateX(*n*)	定义 2D 转换，沿着 X 轴移动元素。
translateY(*n*)	定义 2D 转换，沿着 Y 轴移动元素。
scale(*x*,*y*)	定义 2D 缩放转换，改变元素的宽度和高度。
scaleX(*n*)	定义 2D 缩放转换，改变元素的宽度。
scaleY(*n*)	定义 2D 缩放转换，改变元素的高度。
rotate(*angle*)	定义 2D 旋转，在参数中规定角度。
skew(*x-angle*,*y-angle*)	定义 2D 倾斜转换，沿着 X 和 Y 轴。
skewX(*angle*)	定义 2D 倾斜转换，沿着 X 轴。
skewY(*angle*)	定义 2D 倾斜转换，沿着 Y 轴。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>transform元素变化</title>
  <style type="text/css">
    @keyframes trans{
      from {
        transform: translate(0,0);
      }
```

```
    to{
      transform: translate(150px,0);
    }
  }
  @keyframes scal{
    /*x轴放大一倍，y轴也放大一倍*/
    from {
      transform: scale(1,1);
    }
    to{
      transform: scale(2,2);
    }
  }
  @keyframes rotat{
    /*旋转*/
    from {
      transform: rotate(0deg);
    }
    to{
      transform: rotate(360deg);
    }
  }
  @keyframes skw{
    /*斜切(倾斜)*/
    from {
      transform: skew(0deg , 0);
    }
    to{
      transform: skew(45deg,0);
    }
  }

  .box{
    height: 100px;
    width: 100px;
    background-color: gold;
    border: 1px solid black;
    margin: 50px auto;
    /*transform: translate(30px,0px);*/
    animation: trans 500ms linear alternate infinite;
  }
   .scale{
    height: 100px;
    width: 100px;
    background-color: gold;
    border: 1px solid black;
    margin: 50px auto;
    /*transform: translate(30px,0px);*/
    animation: scal 500ms linear alternate infinite;
  }
  .rotate{
    height: 100px;
    width: 100px;
    background-color: gold;
    border: 1px solid black;
    margin: 50px auto;
    animation: rotat 500ms infinite linear alternate;
```

```
    }

    .skew{
      height: 100px;
      width: 100px;
      background-color: gold;
      border: 1px solid black;
      margin: 50px auto;
      animation: skw 500ms infinite linear alternate;
    }




  </style>
</head>
<body>

  <div class="box"></div>
  <div class="scale"></div>
  <div class="rotate"></div>
  <div class="skew">斜切动画</div>

</body>
</html>
```

animation动画（关键帧动画）

关键帧动画一般和transform变换一起使用，animation动画不需要触发就可以直接执行。

属性	描述
@keyframes	规定动画。
animation	所有动画属性的简写属性，除了 animation-play-state 属性。
animation-name	规定 @keyframes 动画的名称。
animation-duration	规定动画完成一个周期所花费的秒或毫秒。默认是 0。
animation-timing-function	规定动画的速度曲线。默认是 "ease"。
animation-delay	规定动画何时开始。默认是 0。
animation-iteration-count	规定动画被播放的次数。默认是 1。
animation-direction	规定动画是否在下一周期逆向地播放。默认是 "normal"。
animation-play-state	规定动画是否正在运行或暂停。默认是 "running"。

animation动画属性：

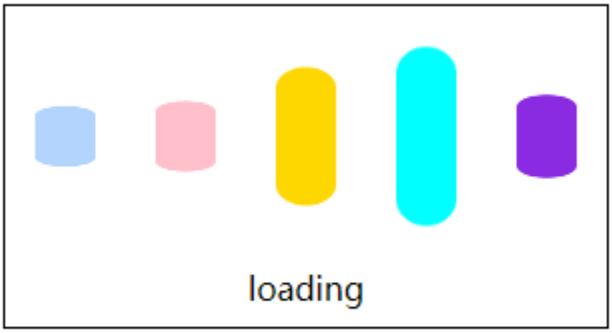
属性	描述
animation-name	用来指定关键帧动画的名字
animation-duration	用来指定动画播放所需的时间，一般以秒为单位
animation-timing-function	设置动画的播放方式
animation-delay	指定动画的开始时间，以秒为单位
animation-iteration-count	指定动画播放的循环次数
animation-direction	控制动画的播放方向
animation-play-state	设置动画的播放状态，播放还是暂停
animation-fill-mode	设置动画的时间外属性

（1）animation-name：none为默认值，将没有任何动画效果，其可以用来覆盖任何动画
（2）animation-duration：默认值为0，意味着动画周期为0，也就是没有任何动画效果
（3）animation-timing-function：与transition-timing-function一样
（4）animation-delay：在开始执行动画时需要等待的时间
（5）animation-iteration-count：定义动画的播放次数，默认为1，如果为infinite，则无限次循环播放
（6）animation-direction：默认为nomal，每次循环都是向前播放，（0-100），另一个值为alternate，动画播放为偶数次则向前播放，如果为基数词就反方向播放
（7）animation-state：默认为running，播放，paused，暂停
（8）animation-fill-mode：定义动画开始之前和结束之后发生的操作，默认值为none，动画结束时回到动画没开始时的状态；forwards，动画结束后继续应用最后关键帧的位置，即保存在结束状态；backwards，让动画回到第一帧的状态；both：轮流应用forwards和backwards规则。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>关键帧动画</title>
  <style type="text/css">

    @keyframes moving {
    /*  定义关键帧动画*/
      from{

      }
      to{
        width: 100px;
        height: 100px;
        border-radius: 50px;
        background-color: pink;
      }
    }

    .box{
      height: 150px;
      width: 150px;
      background-color: red;
      border: 1px solid pink;
      border-radius: 75px;
```

```
      /*alternate ：动画结束后沿原路返回，normal：动画结束后恢复到原状态*/
      animation: moving 1s ease infinite alternate;
      animation-play-state: paused;
    }

    .box:hover{
    /*  设置鼠标放上去后停止动画*/
      animation-play-state: running;
    }
  </style>
</head>
<body>

  <div class="box"></div>

</body>
</html>
```

案例：Loading动画




```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>loading动画</title>
  <style type="text/css">
    .box{
      height: 160px;
      width: 300px;
      border: 1px solid black;
      margin: 50px auto;
    }
    .box div{
      height: 100px;
      width: 30px;
      margin: 15px;
      background: gold;
      float: left;
      border-radius: 15px;
      /*animation: load 500ms ease infinite alternate;*/
    }
    /*css3中新增的选择器，孩子选择器*/
    .box div:nth-child(1){
```

```
      background-color: #b3d4fc;
      animation: load 500ms ease infinite alternate ;

    }
    .box div:nth-child(2){
      background-color: pink;
      animation: load 500ms ease infinite alternate 200ms;
    }
    .box div:nth-child(3){
      background-color: gold;
      animation: load 500ms ease infinite alternate 400ms;
    }
    .box div:nth-child(4){
      background-color: cyan;
      animation: load 500ms ease infinite alternate 600ms;
    }
    .box div:nth-child(5){
      background-color: blueviolet;
      animation: load 500ms ease infinite alternate 800ms;
    }
    .box p{
      text-align: center;
    }

  /*  定义关键帧动画*/
    @keyframes load {
      from{
        /*参数x = 1, y = 1*/
        transform: scale(1,1);
      }
      to{
        transform: scale(1,0.3);
      }
    }

  </style>
</head>
<body>

<div class="box">
  <div></div>
  <div></div>
  <div></div>
  <div></div>
  <div></div>
  <p>loading</p>
</div>

</body>
</html>
```

练习：心跳

流体布局

流体布局，就是使用百分比来设置元素的宽度，元素的高度按实际高度写固定值，流体布局中，元素的边线无法用百分比，可以使用样式中的计算函数 calc() 来设置宽度，或者使用 box-sizing 属性将盒子设置为从边线计算盒子尺寸。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>流体布局</title>

  <style type="text/css">
    .box{
      margin: 0px;
      margin-left: -1px;
    }
    .box a{
      display: block;
      float: left;
      background-color: gold;
      /*width:25% ;*/
      height: 50px;
      line-height: 50px;
      text-align: center;
      text-decoration: none;
    /*  边框尺寸解决方案一：减去边框宽度达到适配效果*/
    /*  width: calc(25% - 2px);*/
    /*  解决方案二：把边框尺寸直接计算到盒子本身宽度内*/
      width: 25%;
      box-sizing: border-box;

    }
  </style>
</head>
<body>

<div class="box">
  <a href="#">首页</a>
  <a href="#">商品主页</a>
  <a href="#">联系我们</a>
  <a href="#">招贤纳士</a>
</div>
</body>
</html>
```

响应式布局

响应式布局就是使用媒体查询的方式，通过查询浏览器宽度，不同的宽度应用不同的样式块，每个样式块对应的是该宽度下的布局方式，从而实现响应式布局。响应式布局的页面可以适配多种终端屏幕（pc、平板、手机）。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>响应式布局</title>
  <style type="text/css">
    body{
      margin: 0;
    }

    .box div{
      width: 23%;
      background-color: plum;
      border: 2px solid black;
      height: 100px;
      box-sizing: border-box;
      text-align: center;
      /*上下左右各1%*/
      margin: 1%;
      float: left;
    }

  /*  媒体查询实现响应式布局*/
    @media (max-width: 800px){
    /*  宽度时400px-800px以下是以下样式*/
      .box div{
        width: 46%;
        margin: 2%;
      }
    }
    /*  媒体查询实现响应式布局*/
    @media (max-width: 400px){
    /*  宽度时400px以下是以下样式*/
      .box div{
        width: 96%;
        margin: 2%;
      }
    }
  </style>
</head>
<body>

<div class="box">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
</div>
```

```
</body>
</html>
```